REMARKS

In the Official Action mailed on **27 November 2006**, the Examiner reviewed claims 1-34. The drawings were objected to as failing to comply with 37 C.F.R. §1.84(p)(5). Claims 12-22 were rejected under 35 U.S.C. §101 because the claimed invention is directed to non-statutory subject matter. Claims 1, 2, 6, 7, 12, 13, 17, 18, 23, 24, 28, and 29 were rejected under 35 U.S.C. §103(a) as being unpatentable over Merrick et al. (USPN 6,339,841 hereinafter "Merrick"), in view of Tock (USPN 5,815,718 hereinafter "Tock"). Claims 3-5, 10, 11, 14-16, 21, 22, 25-27, 32, 33, and 34 were rejected under 35 U.S.C. §103(a) as being unpatentable over Merrick and Tock, and further in view of Chen ("*Java Card Technology for Smart Cards: Architecture and Programmer's Guide*", hereinafter Chen). Claims 8, 9, 19, 20, 30, and 21 were rejected under 35 U.S.C. §103(a) as being unpatentable over Merrick and Tock, in view of Leroy ("*On-Card Bytecode Verification for Java Card*" hereinafter "Leroy").

Objections

The Examiner objected to the drawings for failing to comply with 37 C.F.R. § 1.84(p)(5) because they included a reference character not included in the description. Applicant has amended the specification to add the reference character to the description.

Rejections under 35 U.S.C. §101

The Examiner rejected claims 12-22 under 35 U.S.C. §101 because the claimed invention is directed to non-statutory subject because the computer readable storage medium of claims 12-22 was defined as being "embodied in a transmission medium." Applicant has amended par. [0026] of the disclosure to remove the definition of the computer readable storage medium as a transmission medium.

Rejections under 35 U.S.C. §103(a)

Claims 1, 2, 6, 7, 12, 13, 17, 18, 23, 24, 28, and 29 were rejected under 35 U.S.C. §103(a) as being unpatentable over Merrick in view of Tock. Applicant respectfully points out that Merrick is fundamentally distinct from the present invention because the Merrick system is confined to limiting the number of classes that are loaded during the ClassLoading operation.

In the system taught by Merrick, the normal ClassLoading operation is modified to distribute class data at a "method level" of granularity (see Merrick, col. 2, lines 35-41). In Merrick's system, this results in the following scenario:

> "1) A client program requires the use of a remote class.
>
> 2) A skeleton definition of the class is sent to the client. State metadata and constant pool information is downloaded at this point together with any method identified as always needing to be downloaded.
>
> 3) When a method is actually referenced, lazy verification also triggers the downloading of that fragment of the class file. Only those methods actually referenced are transferred." (see Merrick, col. 2, lines 42-50)

In other words, in Merrick's system, the **byte code for all the methods is not necessarily loaded** during the ClassLoading operation. Instead, some methods are **initially loaded** as part of a minimum class requirement, while the remaining methods are **loaded as needed** using a specially prepared load method invoker (i.e., a modified class loader) (see Merrick col. 4, lines 15-27). In addition, Merrick's system is dependent on other language constructs, such as the constant pool used to define constants used by the class (see Merrick, col. 4, lines 8-15).

In contrast, embodiments of the present invention provide a system that **transforms the method code** as a class is loaded into non-volatile memory (i.e., EEPROM 104) (see par. [0041] - [0043] of the instant application). For example, the system verifies type safety, verifies branch targets, and creates class data structures while loading the class. In addition, the system **resolves byte codes**, as much as possible, when they are loaded into the non-volatile memory (see par. [0042] of the instant application). Because the byte codes are resolved during loading, there is **no need for the constant pool** found in typical Java applications, which can save a great deal of space (see par. [0042] of the instant application). Nothing in Merrick or Tuck, alone or in concert, suggests transforming method code as classes are loaded.

Applicant has amended claims 1, 12, 23, and 34 to clarify that the system transforms the method code as classes are loaded. These amendments find support in par. [0042] of the instant application. No new matter has been added.

Hence, Applicant respectfully submits that independent claims 1, 12, 23, and 34 as presently amended are in condition for allowance. Applicant also submits that claims 2-11, which depend upon claim 1, claims 13-22, which depend upon claim 12, and claims 24-33, which depend upon claim 23, are for the same reasons in condition for allowance and for reasons of the unique combinations recited in such claims.

CONCLUSION

It is submitted that the present application is presently in form for allowance. Such action is respectfully requested.

Respectfully submitted,

By
Shun Yao
Registration No. 59,242

Date: 30 January 2007

Shun Yao
Park, Vaughan & Fleming LLP
2820 Fifth Street
Davis, CA 95618-7759
Tel: (530) 759-1667
Fax: (530) 759-1665
Email: shun@parklegal.com